

Bioconductor's sva package

Jeffrey Leek and John Storey
Department of Biostatistics
University of Washington
email: jtleek@u.washington.edu

June 14, 2007

Contents

1 Overview	1
2 Simulated Example	1
3 The sva function	2
4 The svaplot function	3

1 Overview

The `sva` package contains functions for the identifying and building surrogate variables for high-dimensional data sets. Surrogate variables are covariates constructed directly from high-dimensional data (e.g., gene expression data) that can be used in subsequent analyses (e.g., differential expression analysis). Surrogate variables are designed to overcome the ubiquitous problem of dependence in multiple testing. The use of surrogate variables in differential expression analysis has been shown to reduce dependence, stabilize error rate estimates, and improve reproducibility see [1] for more detailed information.

This document provides a tutorial for using the `sva` package. The package consists of four functions: (`sva` for identifying and building surrogate variables from a data set and a model matrix, `sva.id` for identifying important surrogate variables from the data matrix and a model matrix, `sva.build` constructs a specified number of surrogate variables from the data matrix a matrix of residuals, and `svaplot` for visualizing the results with graphs. As with any R package, detailed information on functions, their arguments and value, can be obtained in the help files. For instance, to view the help file for the function `sva` within R, type `? sva`. Here we will demonstrate the use of `sva` and `svaplot` to analyze a simulated expression experiment.

2 Simulated Example

We demonstrate the functionality of this package using simulated gene expression data that clearly illustrates the characteristics of the SVA approach. The data used in this analysis is included with the `sva` package as the dataset `svadat`. This data set consists of simulated data for 1000 genes (in rows) and 20 arrays (in columns). The first 10 arrays correspond to the first group and the

second 10 correspond to the second. A second factor also affects the gene expression data and is an indicator variable described below. Genes 1-300 are differentially expressed across groups and genes 200-500 are differentially expressed with respect to the second factor. The goal is to identify genes differentially expressed across groups. We show how surrogate variables are estimated and how this affects a differential expression analysis.

To load the data set type `data(svadat)`, and to view a description of this data type ? `svadat`.

```
> library(sva)
> data(svadat)
> dim(svadat)
```

```
[1] 1000  20
```

3 The sva function

The `sva` function computes surrogate variables from the gene expression matrix and model matrix for a microarray experiment. In the expression matrix, genes should be in rows and arrays in columns. It is assumed that the number of genes is much larger than the number of arrays. First we test for differential expression with respect to group, assign the p-values for each gene to the vector `pu` and plot a histogram of the p-values

```
> grp <- rep(c(0, 1), each = 10)
> hfact <- rep(c(0, 1), 10)
> mod <- cbind(rep(1, 20), grp)
> pu <- apply(svadat, 1, function(x) {
+   summary(lm(x ~ grp))$coeff[2, 4]
+ })
> hist(pu)
```

Then we calculate the surrogate variables using the `sva` and assign the result to the surrogate variable object `svaobj`. We can compare the estimated surrogate variable to the second factor and see if they match.

```
> svaobj <- sva(svadat, mod, seed = 1234)
> plot(svaobj$sv[, 1], col = gray(0.3), pch = "x", xlab = "Array",
+   ylab = "Value", ylim = c(-0.3, 0.3))
> points((hfact - 0.5) * 0.48, col = "blue", pch = "x")
> cor(hfact, svaobj$sv[, 1])
```

```
[1] 0.9993035
```

We can also calculate the p-values adjusted for surrogate variables by including the surrogate variables as covariates in the linear model fits. A comparison of the null p-values from each analysis can be performed by examining the histograms, or the Kolmogorov-Smirnov test of equality with the uniform distribution.

```

> mm <- model.matrix(~-1 + svaobj$sv)
> pa <- apply(svadat, 1, function(x) {
+   summary(lm(x ~ grp + mm))$coeff[2, 4]
+ })

> par(mfrow = c(1, 2))
> hist(pu[301:1000], main = "Unadjusted P-values")
> hist(pa[301:1000], main = "Adjusted P-values")

> ks.test(pu[301:1000], "punif")$p.value

[1] 0.000753155

> ks.test(pa[301:1000], "punif")$p.value

[1] 0.3004743

```

4 The svaplot function

The `svaplot` first plots the percent of variation explained by each eigengene in the residual matrix, where the expected value is approximately $\frac{1}{n-df}$ if there is no structure to the residual matrix, where n is the number of arrays and df is the degrees of freedom of the model. Then, for each surrogate variable two plots are created:

1. An indicator plot, indicating which genes were used to estimate the surrogate variable.
2. A plot of the surrogate variable values versus array.

```

> svaplot(svaobj, select = 0)

> svaplot(svaobj, select = 1)

```

References

- [1] J.T. Leek and J.D. Storey. Capturing heterogeneity in gene expression studies by ‘surrogate variable analysis’. *Submitted*, 2007.

Histogram of pu

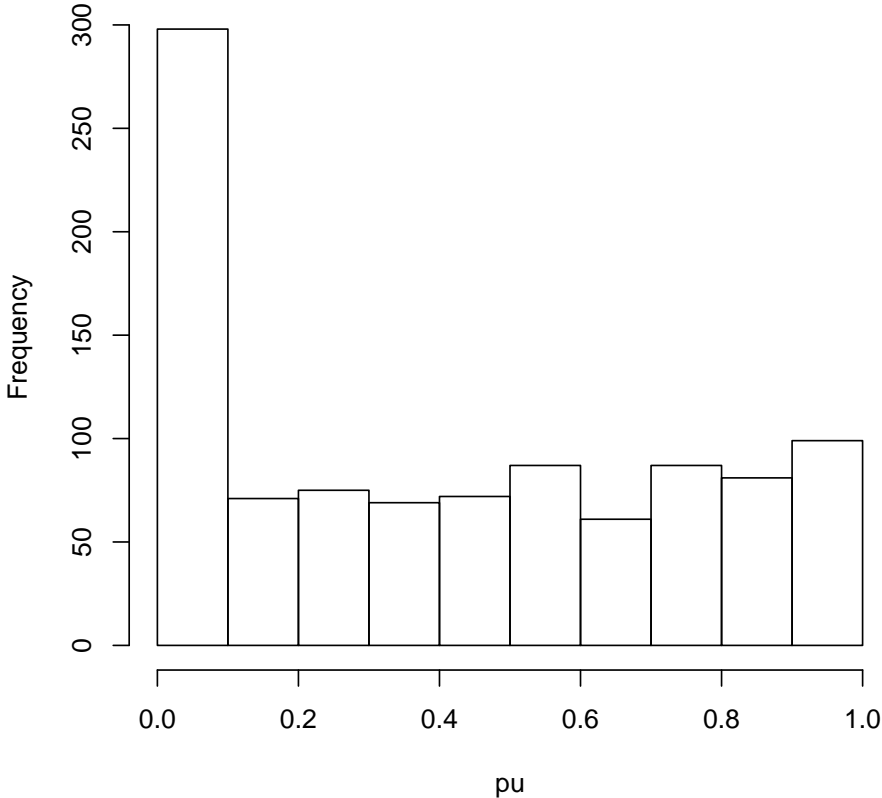


Figure 1: Histogram of unadjusted p-values.

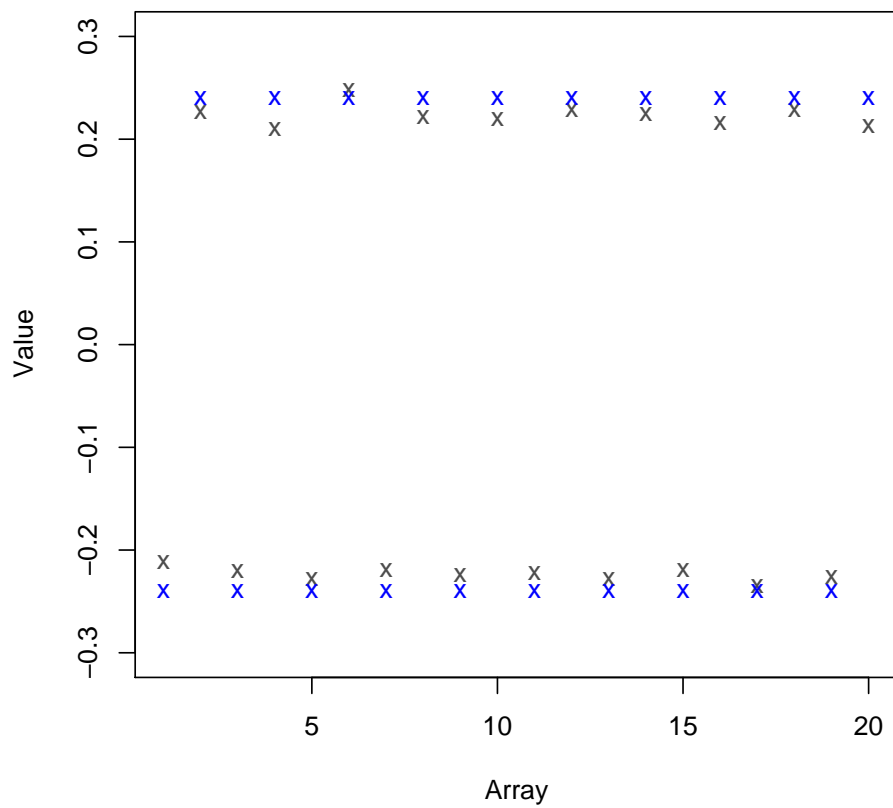


Figure 2: A plot of the estimated surrogate variable (grey) and the true secondary factor (blue).

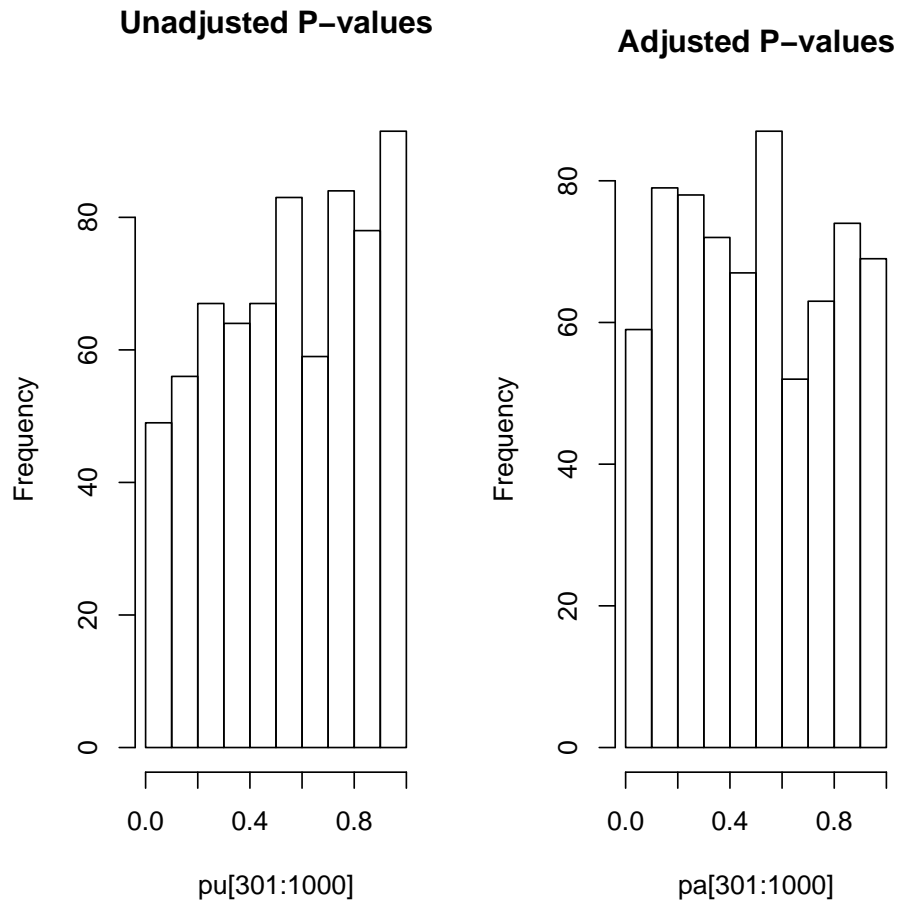


Figure 3: Histograms of the unadjusted and the SVA adjusted p-values from differential expression analysis for group.

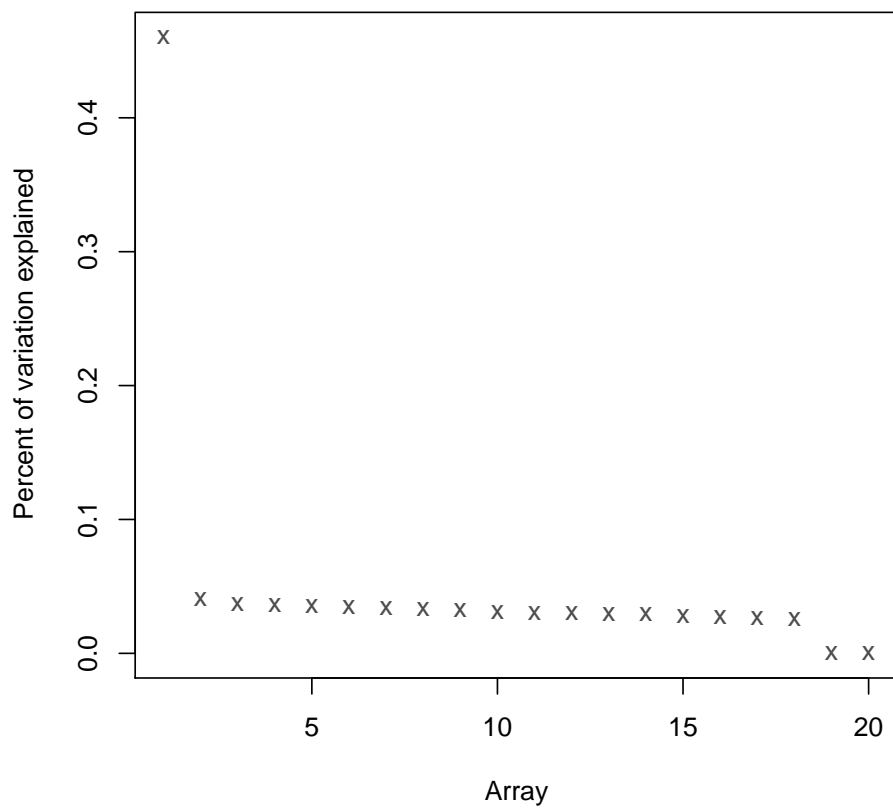


Figure 4: The first plot from `svaplot` showing the percent of variation explained by each eigenvalue in the residual matrix.

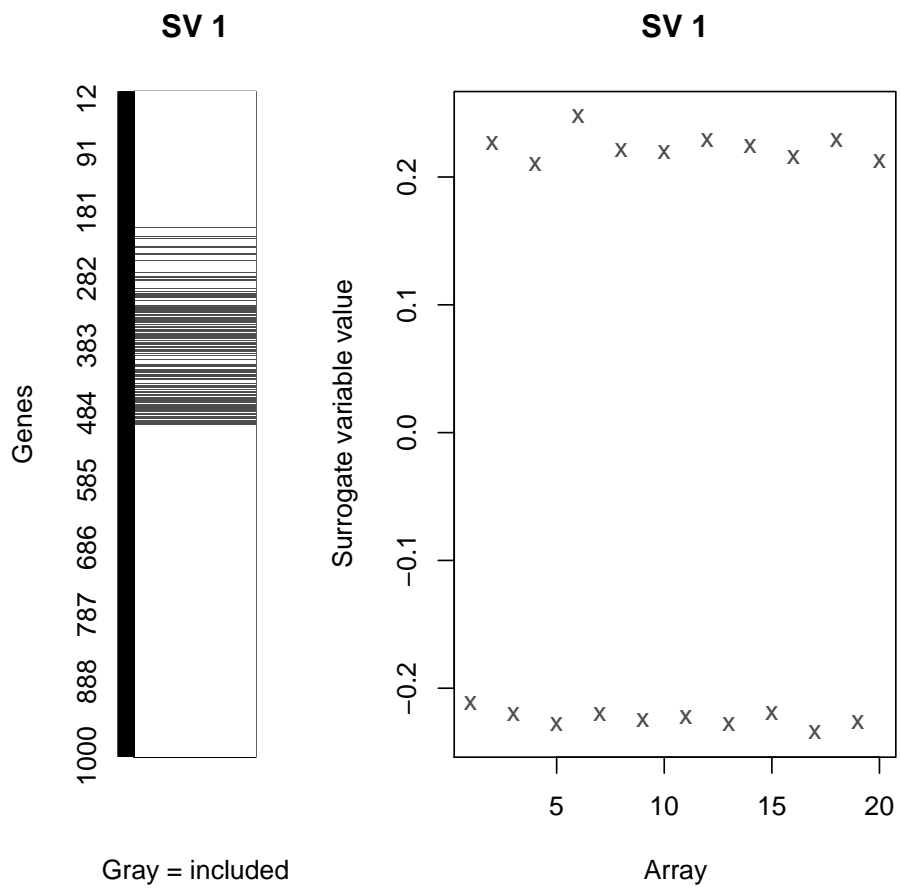


Figure 5: Output from `svaplot` which shows both an indicator of which genes were used in the construction of the first surrogate variable and the surrogate variable values versus arrays.